Length Indexed Bloom Filter Based Forwarding in Content Centric Networking

Vishal Manghnani
Advisor: Dr. Minseok Kwon

Background

Named Data Networking (NDN) is future internet architecture that uses content names instead of host IP for communication. Names are hierarchical and divided into components delimited by “/”, similar to URL or file system names. Name prefixes in NDN forwarding are of variable and unbounded length. This creates scalability issue due to its growing size and longest prefix match becomes a bottleneck as forwarding table size increases.

Bloom Filter is a compact probabilistic data structure that does membership test to determine if the element is present in a set or not. It can be used as a complementary data structure to forwarding table in NDN and can be stored on-chip memory which helps to speed up the process of longest prefix match. However its false positive rate affects the speed.

Abstract

- We introduce a Length indexed bloom filter (LIBF) based tree datastructure for efficient Longest prefix match in Named data network forwarding that enables us to find the longest match given the initial length of prefix. Name prefixes are distributed based on number of components into set of bloom filters arranged in binary tree structure.
- Results show that the LIBF helps in reducing the false positive rate and increasing speed of longest prefix match at the expense of slight increase in memory compared to single bloom filter.

Length Indexed Bloom Filter

- Length Indexed Bloom Filter (LIBF) contains NDN forwarding table name prefixes distributed into set of nodes containing bloom filters based on number of components (length) in each name prefix and arranged in the form of a binary tree.
- Longest Prefix Match (LPM) in LIBF improves from K to log(K), where K is the maximum number of components in NDN name prefixes.
- LPM starts with initial length calculated using the frequency distribution of name prefixes at each node.
- LPM proceeds to right node if the bloom filter in current node contains name prefix of the same length or else it proceeds to left node.
- LPM halts if the search reaches either minimum or maximum possible length.

Benefits

- Supports search from any length
- Provide good average lookup time
- Avoids redundant searches
- Reduces false positive rate

Results

Figure 4 Memory Usage

Figure 5 False positive rates

Figure 6 Number of Hash Probes

Conclusions

- 100,000 name prefixes are generated from file system using “find command” to test the performance of LIBF.
- For simulation NDNSim based on NS-3 is used which simulates the functionality of forwarding in NDN.
- Name prefixes contains maximum 30 components delimited by “/”.
- Memory for LIBF is increased but speed and false positive rates are improved compared to the traditional hash table and single bloom filter.

Future Work

- It will be tested on real routers and the throughput at each router can be calculated.

References


Contact

Vishal Manghnani
Rochester Institute of Technology
Email: vjm1952@rit.edu
Phone: 585-732-0610